**Introduction:** Types of languages – Procedural, Functional, Object Oriented

Object-> code + data

**Static vs Dynamic Languages:**

* **Static:** Type check at compile time
  + Errors will show at compile time
  + More control
  + Declare datatype before you use it
* **Dynamic:** Perform type check at runtime
  + Error might not show till program is run
  + No need to declare data type of variables
  + Might give error at compile time

**Introduction to JAVA:**

.java file -> **compile**-> .class file (byte code) -> **Interpreter** (line by line)->M/C code (0,1)

**Byte Code:**

We need JVM to run this code and converted to machine code.

Can run on all operating system.

**Architecture of Java:**

JDK = JRE + Development tools

JRE = JVM + Library Classes

JVM has JIT

**Static:**

Run function without creating object of the class.

**main:**

Entry point of java code

**String[] args:**

Command line arguments, we can pass these arguments from cmd

java Main “Devarshi”

**Change Bytecode Location:**

javac -d . Demo.java

javac -d .. Demo.java

**Package:**

Folder to locate java files

And to provide access

**Primitive Data Types in Java:**

Any data type which cannot be break into any other data type.

Int a = 10;-> 10 is literal , a is an identifier

**Type Casting:**

Float > integer

If asking for integer but giving float will not work but vice versa will work.

Int num = (int)(67.56f); 🡪 type casting 🡪 67

Ex: int a = 257;

byte b = (byte)(a);

b 🡪 1 (max value of byte is 256 so 1 is remainder of maximum value)

**Java follows UNICODE principle**

**Type promotion rules:**

Two types operation will give bigger data type return value

EX:

byte b = 42;

char c = ‘a’;

short s = 1024;

int I = 50000;

float f = 5.67f

double d = 0.1234;

double result = (f\*b) + (i/c) – (d-s);

* Float value 1777.016

**While loop:**

**For loop:**

[**Q1\_LargestNumber.java**](Codes/Q1_LargestNumber.java)

[**Q2\_AlphabetCaseCheck**](Codes/Q2_AlphabetCaseCheck.java)

[**Q3\_FibonacciNumbers**](Codes/Q3_FibonacciNumbers.java)

**Q4\_CountingOccurrences**